Present Value

One way to analyze a stream of payments or costs that occur over time is to reduce the entries in the series to present value terms. Doing so allows one to compare streams that havedifferent time profiles or different lengths.

1 A Stream of Discrete Payments

This section shows how Maxima can compute the present value of a series of T payments. Each payment is assumed to be made at the end of a specified period. Under that condition, the present value has the following value:

(%i1)
```
sum(x[t]/(1+r)^t,t, 1, T);
```

(%o1)
$$\sum_{t=1}^{T} \frac{x_t}{(r+1)^t}$$

Here x[t] is the payment (or cost) that occurs in period t, where t ranges from 1 to T. If n is specified, then Maxima returns a simple sum:

(%i2)
```
sum(x[t]/(1+r)^t,t, 1, 5);
```

(%o2)
$$\frac{x_1}{r+1}+\frac{x_2}{(r+1)^2}+\frac{x_3}{(r+1)^3}+\frac{x_4}{(r+1)^4}+\frac{x_5}{(r+1)^5}$$

The values of x[t] may be constant or they may vary. If they vary, the variation can be defined by a function or not.

1.1 Present Value, Equal Payments

The next cell defines a present value function for a series of discrete payments of a given size.

(%i3)
```
kill(all)$
pv(x, r, T) := sum(x/(1+r)^t,t,1,T);
```

(%o1) $$pv(x, r, T) := \sum_{t=1}^{T} \frac{x}{(1+r)^t}$$

We can evaluate this function for any given combination of x, r, and n values. For example:

```
(%i2) fpprintprec:5$
      [pv(100,.05,50), pv(100, .10, 50), pv(100, .10, 100)];
      fpprintprec:0$
```

```
(%o3) [1825.6, 991.48, 999.93]
```

Note two features of the results above. First, doubling the interest rate, given that n = 50, reduces pv to just over 1/2 its initial value. Second, doubling n from 50 to 100, given that r = 10%, has only a slight impact on pv.

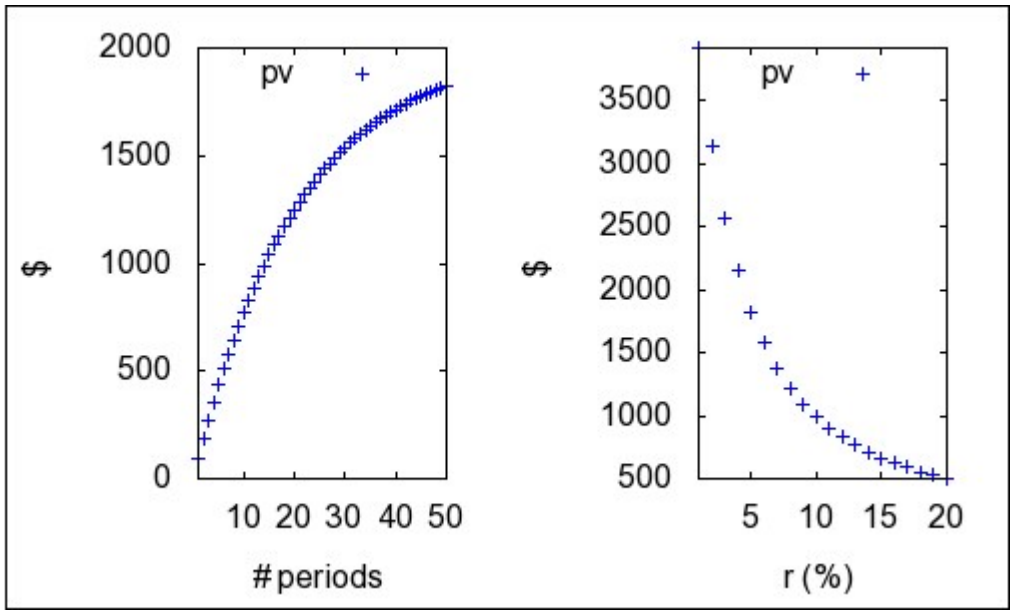
Exercise: Replicate the example, but use n = 5 and n = 10.
Exercise: Replicate the example, but use r = 0.01 and r = 0.02.

A pair of graphs gives a more general picture of how changing r or n affects the present value of a stream of payments. We use gr2d to create two scenarios and wxdraw to plot them.

```
(%i5) pvnList: makelist(pv(100,0.05,n),n,1,50)$
      pvrList:makelist(pv(100,R/100.0, 50),R,1,20 )$
```

```
(%i7) StreamLength: gr2d(xlabel="# periods",
      ylabel="$", key = "pv", xtics=10,ytics=500,
      yrange = [0,2000], points(pvnList))$
      DiscountRate: gr2d(xlabel="r (%)",ylabel="$",
      xtics=5,
      key = "pv", points(pvrList)
      )$
      wxdraw(StreamLength, DiscountRate, columns=2)$
```

(%t9)



Exercise: In the input cell that creates the lists, create a list named pvnList2 with r = 0.10 (R = 10). Add an option points(pvList2) to the StreamLength scenario. Between the first and second points( ) entries, add color=red and key="pv2". Then create a second list PVrList2 that changes n from 50 25. Make the appropriate changes to the DiscountRate scenario. Then run wxdraw once more. Discuss the changes. Be sure to put commas where they belong in the gr2d() commands.

Why not use the following alternative, rather than bothering with creating and printing lists? Answer: the wxdraw step will take a long time to calculate, because Maxima treats t as a continuous variable. This treatment requires that draw very many calculated values in order to construct apparently continuous curves.

The command brackets /* */ keep Maxima from executing this set of commands. An input cell cannot end with */, so we add "end;".

```
(%i10)  /*
        Length: gr2d(xlabel = "T", ylabel="$", key="pv",
          explicit(pv(100, 0.05, T), T, 1, 20 ) )$
        DiscountRate: gr2d(xlabel = "r", ylabel="$", key="pv",
          explicit(pv(100,r, 20), r, .001, .150 ) )$
        wxdraw(Length, DiscountRate)$ */ end;

(%o10)  end
```

1.2 Present Value, Variable Payments

The cell below shows how to compute the present value the payments that constitute a stream that can differ both in size and periodicity. [With only a small addition, an interest rate list, one could allow for interest rates to vary among periods.] In the illustrative example five payments are to be received at the end of 1, 5, 10, 15, and 25 periods. These values appear in the list named tList. Then pList shows the payment size.

The list of present values, pvList, is created by applying the present value formula pv = p/(1 + r)^t individually for each of the five payments.

The makelist command refers to two previous lists, xList and tList. It uses the payments from the former and the time periods from the latter to apply the present value formula. The result is a third list, pvList. For ease of reading, the lists are collected into a matrix named data. A matrix named titles is created by placing a list of titles into a matrix and transposing that matrix. The matrix titles is added to the matrix data using the addcol (add column) command. The resulting table appears as the only piece of output.

```
(%i11) fpprintprec:5$
       tList : [1, 5, 10, 15, 25]$
       xList : [-100, 200, 300, 200, 300]$
       pvList: makelist(xList[i]/(1 + .05)^tList[i], i, 1, 5 )$
       data: matrix(tList,xList, pvList)$
       titles : transpose(matrix(["Time","Net Payment", "PV"]))$
       addcol(titles,data); fpprintprec:0$
```

(%o17)

Time	1	5	10	15	25
Net Payment	-100	200	300	200	300
PV	-95.238	156.71	184.17	96.203	88.591

Now we compute the present value of these five net payments.

Exercise: Hand calculate the PV entries above to ensure that the value below is correct, except for rounding.

```
(%i19) sum(pvList[i],i,1,5);
```

```
(%o19) 430.4353652453469
```

1.3 More Frequent Discounting

The first expression below allows for discounting more than once per period. Suppose that quarterly discounting is used; then, m = 4. Monthly discounting would be accomplished with m = 12, and so forth. The second expression shows the result of discounting m times per period when p is received each period.

```
(%i20) 1/(1+r/m)^(m*t);
```

(%o20)

$$\frac{1}{\left(\frac{r}{m}+1\right)^{m\,t}}$$

We apply this formula for p = 100, r = 0.05, and n = 50 with semiannual, quarterly, and daily discounting. Doing so requires that we redefine the present value function. The first line contains housekeeping commands. The first kills the previously-used present value formula. The second shows that the list of functions is now empty. The third creates the new formula.

```
(%i21) kill(pv)$ functions;
       pv(x, r, T, m) := sum(x/(1+r/m)^(m*t),t,1, T);
```

```
(%o22) []
```

$$(\%o23)\quad pv(x, r, T, m) := \sum_{t=1}^{T} \frac{x}{\left(1+\frac{r}{m}\right)^{m\,t}}$$

The next cell applies the new formula to annual, semi-annual, quarterly, and daily discounting (daily is approximate because of leap years). As one would expect, more frequent compounding reduces the present value of this stream of payments.

```
(%i24) [pv(100, 0.05, 50, 1), pv(100, 0.05, 50, 2),
        pv(100, 0.05, 50, 4),pv(100, 0.05, 50, 365)];
```

```
(%o24) [1825.592546055238, 1808.103963677977, 1799.248294860915,
1790.414997910048]
```

We now take compounding frequency to its limit, allowing for instantaneous compounding. The first command below determines the limit of the expression 1/(1+r/m)^(m*t) as m becomes very large. The result is e^(-r*t) = 1/e^(r*r), where e is the base for common logarithms. (Maxima names this constant %e.)

These alternative ways of stating this result can be handy: %e^(-r*t) = 1/%e^(r*t) = exp(-r*t) = 1/exp(r*t).

```
(%i25) limit(1/(1 + r/m)^(m*t), m, inf);
```

$$(\%o25)\quad \%e^{-r\,t}$$

The equation pvi(x,r,T) incorporates this limiting caseof instantaneous compounding. Evaluating it for the values of x, r, and T that we use above shows that the effect of compounding instantaneously rather than daily is slight.

```
(%i26) pvi(x,r,T):= sum(x/exp(r*t), t, 1, T);
       pvi(100,0.05,50);
```

(%o26) $$pvi(x, r, T) := \sum_{t=1}^{T} \frac{x}{\exp(r\,t)}$$

(%o27) 1790.316701332228

When the number of payments is infinite (as in British consols (annuities that continue forever) of the 19th century, or a renewable resource), the present value of the series as returned by the pvi(x,r, inf) equation is not helpful. Nor do we gain anything by evaluating thelimiting case of pv(x, r, T). The sum command correctly recognizes that, strictly speaking, such a sum continues to grow indefinitely. What it does not take into account is that, given theparameter values that we sue, the present value of payments in the far future become small enough to be ignored.

(%i28)
```
pvi(x,r,inf); limit(pvi(x,r,T),T,inf);
```

(%o28) $$\left(\sum_{t=1}^{\infty} \%e^{-r\,t}\right) x$$

(%o29) $$\left(\lim_{T\to\infty} \sum_{t=1}^{T} \%e^{-r\,t}\right) x$$

The alternative nusum command provides a way to analyze the present value of an infinite series of payments. Output %o3 shows the expression of the sum of such a series. Using the part command, the first part of this is retrieved and the second part ignored, as is appropriate for very large values of T. [Rewrite the second part as x/((exp(r*T)*(exp(r) -1). The value of x is a finite constant, the value of exp(r*T) approaches infinity, and the value of exp(r) - 1 is approximately r. Therefore, this second termis extremely small.] This expression is then rephrased as the function pv_inf, which expresses the present value as a function of p and r.

(%i30)
```
nusum(x/exp(r*t),t,1,T );
part(%,1);
pv_inf(x,r) := "%;
```

(%o30) $$\frac{x}{\%e^{r}-1} - \frac{x\,\%e^{-r\,T}}{\%e^{r}-1}$$

(%o31) $$\frac{x}{\%e^{r}-1}$$

(%o32) $$pv\_inf(x, r) := \frac{x}{\%e^{r}-1}$$

For x = 100 and r = 0.05 per year, the resulting value is approximately 1950.42, about 9 percent larger than thepresent value of the stream when T = 50.

```
(%i33) pv_inf(100,0.05);
       %/pvi(100,0.05,50);

(%o33) 1950.416649306586
(%o34) 1.089425489833851
```

2 Continuous Payments

Suppose that payments occur continously through a period. In our example above, suppose that $100 is paid in a very large number of payments such that one receives 100 in each year. We can no longer use the sum command to determine the present value of the stream. We can, however, use its continuous conterpart, integrate. The command below shows that if we integrate 100 over time, from 0 (today) to 1 (the end of the first period/beginning of the second period), the result is 100, as required.

```
(%i35) integrate(100,t, 0,1);

(%o35) 100
```

2.1 Discounting the Payments

Suppose now that instantaneous compounding is applied to this series. Then the present value of the payments is approximately 97.54.

```
(%i36) ratprint:false$
       integrate(100.0/exp(0.05*t), t, 0, 1);

(%o37) 97.54115099856945
```

Compare this value to the present value of 100 received at the end of the period. That present value is just over 95.1, whether compounding occurs once or is instantaneous. Not surprisingly, receiving the income earler increases its present value.

```
(%i38) 100/(1.05); 100/exp(0.05);

(%o38) 95.23809523809524
(%o39) 95.12294245007139
```

The generalization to a larger number of periods is direct. Suppose, that we look

forward to 50 periods of continuous income. The present value is 1835.83, about 2.5 percent larger than the present value of the stream of 50 end-of-period payments.

```
(%i40) integrate(100.0/exp(0.05*t),t, 0, 50);
       %/pvi(100,0.05, 50);

(%o40) 1835.830002752202
(%o41) 1.025421927520481
```

2.2 Variable Payments

One attraction of the use of a continuous representation of the present value of a stream is that, if the stream's values follow a mathematical formula, then calculation becomes quite simple. Suppose the quadratic relationship below represents an income flow. We solve the equation for y(t) = 0 and call for a floating-point value, so we can determine a range over which to apply the function.

```
(%i42) y(t) := -100 + 20.0*t - 0.01*t^2;
       float( solve(y(t),t) );

(%o42) y(t):=-100+20.0 t+(-0.01) t^2
(%o43) [t=5.01256289338005, t=1994.98743710662]
```

Suppose that the function above depicts a flow for 1000 periods. Then the present value of that flow, with r = 5%, is approximately 6,566,667.

```
(%i44) integrate(y(t),t,0,1000); float(%);

                19700000
(%o44)          --------
                   3
(%o45) 6566666.666666667
```

An important case of a stream of payments being defined by a function is that of a growth function. Suppose that the payments grow at an annual rate g with instantaneous compounding. That function is defined below.

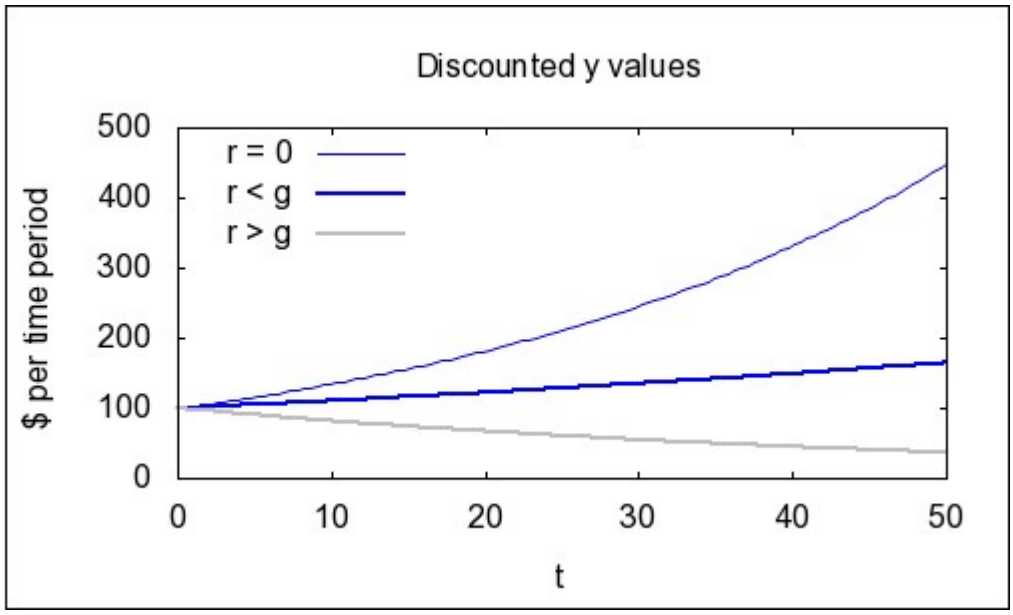
```
(%i46) kill(y)$ y(y0, g, t) := y0*%e^(g*t);

                                 g t
(%o47) y(y0, g, t):=y0 %e
```

The corresponding present value of such a flow is this: y(y0,g,t)/exp(r*t). The graph below shows three profiles. The first is the flow of payments, not discounted. The second shows a flow of present values when r < g. The third shows a flow of

present values when r > g.

```
(%i48) wxdraw2d(title="Discounted y values",
       user_preamble="set key top left",
       xlabel="t", ylabel = "$ per time period",
       yrange = [0, 500],key = "r = 0",
       explicit(y(100,0.03,t), t, 0, 50),line_width=2,
       key = "r < g",explicit(y(100,0.03,t)/exp(0.02*t), t, 0, 50),
       color=gray, key = "r > g",
       explicit(y(100,0.03,t)/exp(0.05*t), t, 0, 50) )$
```

(%t48)



In order to determine the present value of a stream of individual present values requires summation. Because these functions are continuous, however, we replace sum with integrate. The first command below shows the integral for a flow that begins at a value y0 and grows for T periods at a rate g.

The second command produces the integral for the discounted sum. To see that the two are more similar than they might appear, we rearrange the terms. The first expression can be written as y0*(exp(g*T -1))/g, and the second can be rewritten as y0*(exp((g-r)*T)-1)/(g-r). Clearly the first expression is a special case of the second, when r = 0.

The third command uses the quote-quote (" -- two single quotes) operator to convert the second expression to a functional relationship.

```
(%i49) integrate(y(y0,g,t),t,0,T);
       integrate(y(y0,g,t)/exp(r*t),t,0,T);
       npv(y0,g,r,T):= "%;
```

(%o49) $y0\left(\frac{\%e^{g\,T}}{g}-\frac{1}{g}\right)$

(%o50) $y0\left(\frac{1}{r-g}-\frac{\%e^{g\,T-r\,T}}{r-g}\right)$

(%o51) $\mathrm{npv}(y0,g,r,T):=y0\left(\frac{1}{r-g}-\frac{\%e^{g\,T-r\,T}}{r-g}\right)$

We can see that this function can be applied to the example used above, when a stream of x = 100 is received. In that case g = 0, and r = 0.05.

```
(%i52) npv(100,0.0,0.05,50);

(%o52) 1835.830002752202
```

We can determine the areas under each of the three curves above.

```
(%i53) npv(100,0.03,0.0,50);
       npv(100,0.03,0.02,50);
       npv(100,0.03,0.05,50);

(%o53) 11605.63023446022
(%o54) 6487.212707001284
(%o55) 3160.602794142788
```

Finally, we can determine the limit of the npv function as T approaches infinity. When r < g, the npv is a very large number (the computer's approximation to infinity), because the effect of discounting is not enough to offset the effect of growth. When r = 5% and g = 3%, however, extending the time horizon from T = 50 to T = infinity increases npv by only about 58 percent (5000.0 vs. 3160.6, approximately).

```
(%i56) npvLimit: limit(npv(y0,g,r,T),T, 10000);
       ev(npvLimit, g = 0.03, r = 0.02, y0 = 100);
       ev(npvLimit, g = 0.03, r = 0.05, y0 = 100);
       %/3160;
```

(%o56) $\frac{\%e^{-10000\,r}\left(\%e^{10000\,r}-\%e^{10000\,g}\right)y0}{r-g}$

(%o57) $2.6881171418161358\ 10^{47}$

```
(%o58) 4999.999999999999
(%o59) 1.582278481012658
```

When we give Maxima the function and specific values of r and g, Maxima recognizes the case of r < g as one that produces an analytical (vs. numerical)

solution and returns inf (positive infinity) as the value. [wxMaxima shows the infinity symbol; Maxima represents infinities as follows: inf is positive infinity, -inf is negative infinity, and infinity is an infinite value that Maxima cannot sign.]

```
(%i60)  limit(npv(100,0.03,0.02,T),T, inf);
```

```
(%o60) ∞
```
